4

Memory-Efficient Hardware Architecture of 2-D Dual-Mode Lifting-Based Discrete Wavelet Transform for JPEG2000

Chih-Hsien Hsia and Jen-Shiun Chiang
Department of Electrical Engineering, Tamkang University
Taipei, Taiwan

1. Introduction

Discrete wavelet transform (DWT) has been adopted in a wide range of applications, including speech analysis, numerical analysis, signal analysis, image coding, pattern recognition, computer vision, and biometrics (Mallat, 1989). It can be considered as a multi-resolution decomposition of a signal into several components with different frequency bands. Moreover, DWT is a powerful tool for signal processing applications, such as JPEG2000 still image compression, denoising, region of interest, and watermarking. For real-time processing it needs small memory access and low computational complexity. Implementations of two-dimensional (2-D) DWT can be classified as convolution-based operation (Mallat, 1989) (Marino, 2000) (Vishwanath et al., 1995) (Wu. & Chen, 2001) and lifting-based operation (Sweldens, 1996). Since the convolution-based implementations of DWT have high computational complexity and large memory requirements, lifting-based DWT has been presented to overcome these drawbacks (Sweldens, 1996) (Daubechies & Sweldens, 1998). The lifting-based scheme can provide low-complexity solutions for image/video compression applications, such as JPEG2000 (Lian et al., 2001), Motion-JPEG2000 (Seo & Kim, 2007), MPEG-4 still image coding, and MC-EZBC (Ohm, 2005) (Chen & Woods, 2004). However, the real-time 2-D DWT for multimedia application is still difficult to achieve. Hereafter, efficient transformation schemes for real-time application are highly demanded. Performing 2-D (or multi-dimensional) DWT requires many computations and a large block of transpose memory for storing intermediate signals with long latency time. This work presents new algorithms and hardware architectures to improve the critical issues in 2-D dual-mode (supporting 5/3 lossless and 9/7 lossy coding) lifting-based discrete wavelet transform (LDWT). The proposed 2-D dual-mode LDWT architecture has the merits of low transpose memory, low latency, and regular signal flow, making it suitable for VLSI implementation. The transpose memory requirement of the $N \times N$ 2-D 5/3 mode LDWT is $2N$ and that of 2-D 9/7 mode LDWT is $4N$.

Low transpose memory requirement is of a priority concern in spatial-frequency domain implementation. Generally, raster scan signal flow operations are popular in $N \times N$ 2-D DWT, and under this approach the memory requirement ranges from $2N$ to N^2 (Diou et al., 2001)

(Andra et al., 2002) (Chen & Wu, 2002) (Chen, 2002) (Chiang & Hsia, 2005) (Jung & Park, 2005) (Vishwanath et al., 1995) (Huang et al., 2005) (Mei et al., 2006) (Huang et al, 2005) (Wu & Lin, 2005) (Lan ., 2005) (Wu. & Chen, 2001) in 2-D 5/3 and 9/7 modes LDWT. In order to reduce the amount of the transpose memory, the memory access must be redirected. In our approach, the signal flow is revised from row-wise only to mixed row- and column-wise, and a new approach, called interlaced read scan algorithm (IRSA), is used to reduce the amount of the transpose memory. By the IRSA approach, a transpose memory size is of 2*N* or 4*N* (5/3 or 9/7 mode) for an *N*×*N* DWT. The proposed 2-D LDWT architecture is based on parallel and pipelined schemes to increase the operation speed. For hardware implementation we replace multipliers with shifters and adders to accomplish high hardware utilization. This 2-D LDWT has the characteristics of high hardware utilization, low memory requirement, and regular signal flow. A 256×256 2-D dual-mode LDWT was designed and simulated by VerilogHDL, and further synthesized by the Synopsys design compiler with TSMC 0.18μm 1P6M CMOS process technology.

2. Survey of 2-D LDWT Architecture

Among the variety of DWT algorithms, LDWT provides a new approach for constructing biorthogonal wavelet transforms and also provides an efficient scheme for calculating classical wavelet transforms (Sweldens, 1996) (Chen & Wu, 2002) (Andra et al., 2000) (Andra et al., 2002) (Diou et al., 2001) (Chen, 2002) (Chiang & Hsia, 2005) (Tan & Arslan, 2001) (Huang et al., 2005) (Huang et al., 2002) (Mei et al., 2006) (Weeks & Bayoumi, 2002) (Varshney et al., 2007) (Huang et al., 2004) (Tan & Arslan, 2003) (Jiang & Ortega, 2001) (Jung & Park, 2005) (Chen, 2004) (Lian et al, 2001) (Seo & Kim, 2007) (Huang et al., 2005) (Wu & Lin, 2005) (Lan et al., 2005) (Wu. & Chen, 2001). Factoring the classical wavelet filter into lifting steps can reduce the computational complexity of the corresponding DWT by up to 50% (Daubechies & Sweldens, 1998). The lifting steps can be implemented easily, which is different from the direct finite impulse response (FIR) implementations of Mallat's algorithm (Daubechies & Sweldens, 1998). Andra *et al.* (Andra et al., 2000) (Andra et al., 2002) proposed a block-based simple four-processor architecture that computes several stages of the DWT at a time. Diou *et al.* (Diou et al., 2001) presented an architecture that performs LDWT with a 5/3 filter by interleaving technique. Chen *et al.* (Chen & Wu, 2002) proposed a folded and pipelined architecture for a 2-D LDWT implementation, with memory size of 2.5*N* for an *N*×*N* 2-D DWT. This lifting architecture for vertical filtering is divided into two parts, each consisting of one adder and one multiplier. Since both parts are activated in different cycles, they can share the same adder and multiplier to increase the hardware utilization and reduce the latency. However, this architecture also has high complexity due to the characteristics of the signal flow. Chen *et al.* (Chen, 2002) proposed a flexible folded architecture for 3-level 1-D LDWT to increase the hardware utilization. Chiang *et al.* (Chiang & Hsia, 2005) proposed a 2-D DWT folded architecture to improve the hardware utilization. Jiang *et al.* (Jiang & Ortega, 2001) presented a parallel processing architecture that models the DWT computation as a finite state machine and efficiently computes the wavelet coefficients near the boundary of each segment of the input signal. Lian *et al.* (Lian et al, 2001) and Chen *et al.* (Chen, 2004) used a 1-D folded architecture to improve the hardware utilization of 5/3 and 9/7 filters. The recursive architecture is a general scheme to implement any wavelet filter that is decomposed into lifting steps in

smaller hardware complexity. Jung *et al.* (Jung & Park, 2005) presented an efficient VLSI architecture of dual-mode LDWT that is used by lossy or lossless compression of JPEG2000. Marino (Marino, 2000) proposed a high-speed/low-power pipelined architecture for the direct 2-D DWT by four-subband transforms performed in parallel. The architecture of (Huang et al., 2002) implements 2-D DWT with only transpose memory by using recursive pyramid algorithm (PRA). In (Vishwanath et al., 1995) it has the average of N^2 computing time for all DWT levels. However, they use many multipliers and adders. Varshney *et al.* (Varshney et al., 2007) presented energy efficient single-processor and fully pipelined architectures for 2-D 5/3 lifting-based JPEG2000. The single processor performs both row-wise and column-wise processing simultaneously to achieve the 2-D transform with 100% hardware utilization. Tan *et al.* (Tan & Arslan, 2003) presented a shift-accumulator arithmetic logic unit architecture for 2-D lifting-based JPEG2000 5/3 DWT. This architecture has an efficient memory organization, which uses a small amount of embedded memory for processing and buffering. Those architectures achieve multi-level decomposition using an interleaving scheme that reduces the size of memory and the number of memory accesses, but have slow throughput rates and inefficient hardware utilization. Seo *et al.* (Seo & Kim, 2007) proposed a processor that can handle any tile size, and supports both 5/3 and 9/7 filters for Motion-JPEG2000. Huang *et al.* (Huang et al, 2005) proposed a generic RAM-based architecture with high efficiency and feasibility for 2-D DWT. Wu *et al.* (Wu & Lin, 2005) presented a high-performance and low-memory architecture to implement a 2-D dual-mode LDWT. The pipelined signal path of their architecture is regular and practical. Lan *et al.* (Lan et al., 2005) proposed a scheme that can process two lines simultaneously by processing two pixels in a clock period. Wu *et al.* (Wu. & Chen, 2001) proposed an efficient VLSI architecture for direct 2-D LDWT, in which the poly-phase decomposition and coefficient folding are adopted to increase the hardware utilization. Despite these efficient improvements to existed architectures, further improvements in the algorithm and architecture are still needed. Some VLSI architectures of 2-D LDWT try to reduce the transpose memory requirements and communication between the processors (Chen & Wu, 2002) (Andra et al., 2000) (Andra et al., 2002) (Diou et al., 2001) (Chen, 2002) (Chiang & Hsia, 2005) (Tan & Aslan, 2002) (Jiang & Ortega, 2001) (Lian et al., 2001) (Jung & Park, 2005) (Chen, 2004) (Huang et al., 2005) (Daubechies & Sweldens, 1998) (Marino, 2000) (Vishwanath et al., 1995) (Taubman & Marcellin, 2001) (Marcellin et al., 2000) (Mei et al., 2006) (Varshney et al., 2007) (Huang et al., 2004) (Tan & Arslan, 2003) (Seo & Kim, 2007) (Huang et al., 2005) (Wu & Lin, 2005) (Lan et al., 2005) (Wu. & Chen, 2001), however these hardware architectures still need large transpose memory.

3. Discrete wavelet transform and lifting-based method

This section briefly reviews the use of DWT in the coding engine of JPEG2000 (Taubman & Marcellin, 2001). The classical DWT employs filtering and convolution to achieve signal decomposition (Mallat, 1989) (Marino, 2000) (Vishwanath et al., 1995) (Wu. & Chen, 2001). Meyer and Mallat found that the orthonormal wavelet decomposition and reconstruction can be implemented in the multi-resolution signal analysis framework (Mallat, 1989). The multi-resolution analysis is now a standard method for constructing the orthonormal wavelet-bases. JPEG2000 adopts this characteristic to transform an image in the spatial domain into the frequency domain.

3.1 Classical DWT

DWT performs multi-resolution decomposition of the input signals (Mallat, 1989). The original signals are first decomposed into two subspaces, called the low- (low-pass) and high-frequency (high-pass) subbands. The classical DWT implements the decomposition (analysis) of a signal by a low-pass digital filter H and a high-pass digital filter G. Both digital filters are derived using the scaling function and the corresponding wavelets. The system downsamples the signal to decimate half of the filtered results in decomposition processing. The Z-transfer functions of $H(z)$ and $G(z)$ based on four-tap and non-recursive FIR filters with length L are represented as follows:

$$H(z)= h_0 + h_1z^{-1} + h_2z^{-2}+ h_3z^{-3}, \tag{1}$$

$$G(z)= g_0 + g_1z^{-1} + g_2z^{-2}+ g_3z^{-3}. \tag{2}$$

The reconstruction (synthesis) process is implemented using an up-sampling process. Mallat's tree algorithm or pyramid algorithm (Mallat, 1989) can be used to find the multi-resolution decomposition DWT. The decomposition DWT coefficients at each resolution level can be calculated as follows:

for (j=1 to J)
for (i=0 to $N/2^j$-1)
{

$$X_H^j(n)=\sum_{i=0}^{k-1} G(z)X^{J-1}H(2n-i) \tag{3}$$

$$X_L^j(n)=\sum_{i=0}^{k-1} H(z)X^{J-1}G(2n-i) \tag{4}$$

}

where j denotes the current resolution level, k the number of the filter tap, $X_H^j(n)$ the nth high-pass DWT coefficient at the jth level, $X_L^j(n)$ the nth low pass DWT coefficient at the jth level, and N the length of the original input sequence. Fig. 1 shows a 3-level 1-D DWT decomposition using Mallat's algorithm.

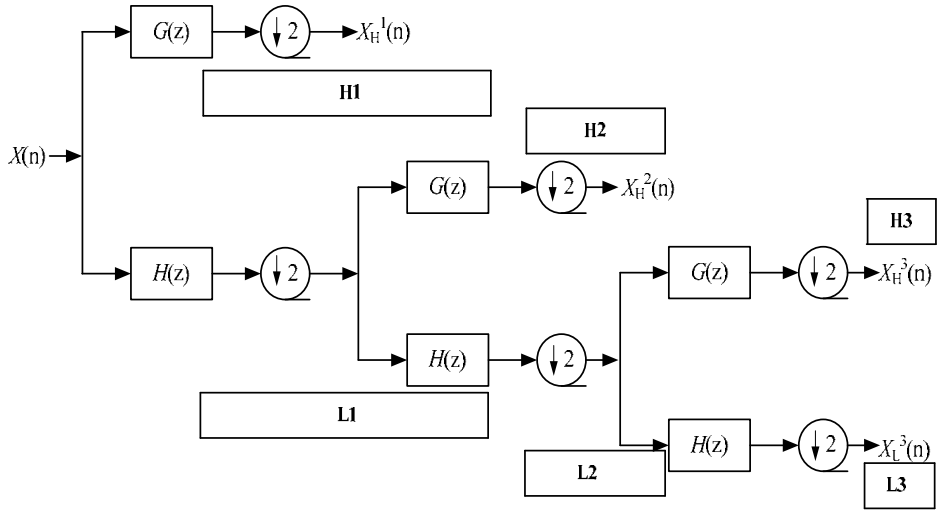


Fig. 1. 3-level 1-D DWT decomposition using Mallat's algorithm.

The downsampling operation is then applied to the filtered results. A pair of filters are applied to the signal to decompose the image into the low-low (LL), low-high (LH), high-low (HL), and high-high (HH) wavelet frequency bands. Fig. 2 illustrates the basic 2-D DWT operation and the transformed result which is composed of two cascading 1-D DWTs. The image is first analyzed horizontally to generate two subimages. The information is then sent into the second 1-D DWT to perform the vertical analysis to generate four subbands, and each with a quarter of the size of the original image. Considering an image of size $N \times N$, each band is subsampled by a factor of two, so that each wavelet frequency band contains $N/2 \times N/2$ samples. The four subbands can be integrated to generate an output image with the same number of samples as the original one.

Most image compression applications can reapply the above 2-D wavelet decomposition repeatedly to the LL subimage, each time forming four new subband images, to minimize the energy in the lower frequency bands.

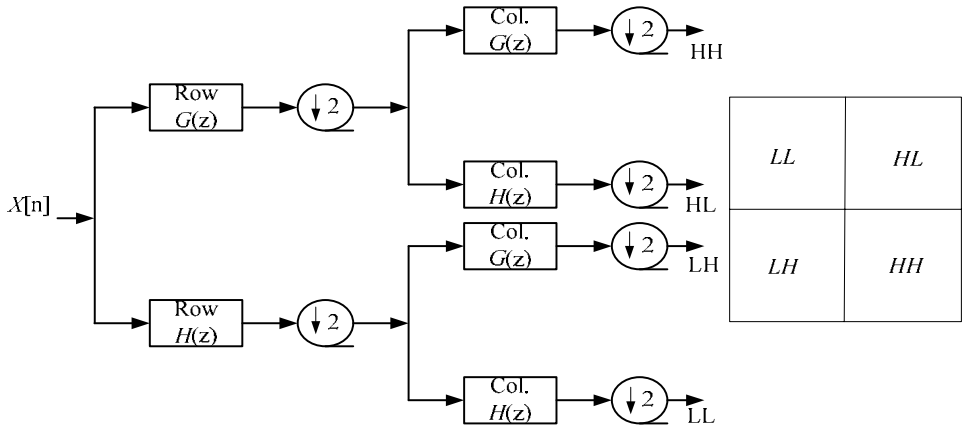


Fig. 2. The 2-D analysis DWT image decomposition process.

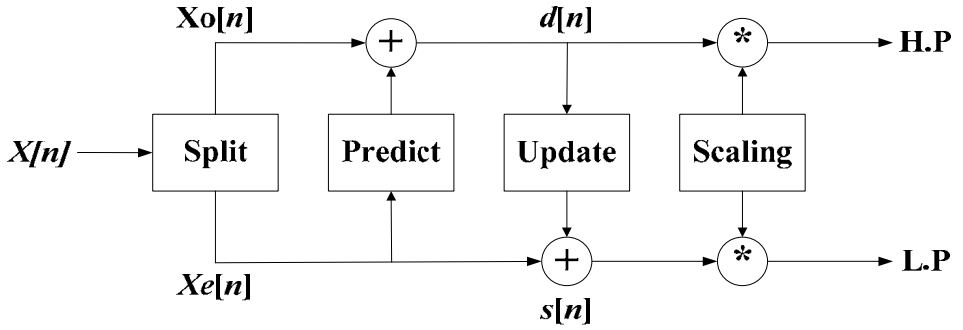


Fig. 3. Block diagram of the LDWT.

3.2 LDWT algorithm

The lifting-based scheme proposed by Daubechies and Sweldens requires fewer computations than the traditional convolution-based approach (Sweldens, 1996) (Daubechies & Sweldens, 1998). The lifting-based scheme is an efficient implementation for DWT; it can easily use integer operations and avoid the problems caused by the finite precision or rounding. The Euclidean algorithm can be used to factorize the poly-phase matrix of a DWT filter into a sequence of alternating upper and lower triangular matrices and a diagonal matrix. The variables $h(z)$ and $g(z)$ in (5) respectively denote the low- and high-pass analysis filters, which can be divided into even and odd parts to generate a poly-phase matrix $P(z)$ as in (6).

$$g(z)=g_e(z^2)+ z^{-1}g_o(z^2),$$

$$h(z)=h_e(z^2)+z^{-1}h_o(z^2). \quad (5)$$

$$P(z) = \begin{bmatrix} h_e(z) & g_e(z) \\ h_o(z) & g_o(z) \end{bmatrix}. \tag{6}$$

The Euclidean algorithm recursively finds the greatest common divisors of the even and odd parts of the original filters. Since h(z) and g(z) form a complementary filter pair, P(z) can be factorized into (7):

$$P(z) = \prod_{i=1}^{m} \begin{pmatrix} 1 & s_i(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ t_i(z) & 1 \end{pmatrix} \begin{pmatrix} k & 0 \\ 0 & 1/k \end{pmatrix}. \tag{7}$$

where s_i(z) and t_i(z) are Laurent polynomials corresponding to the prediction and update steps, respectively, and k is a nonzero constant. Therefore, the filter bank can be factorized into three lifting steps.

As illustrated in Fig. 3, a lifting-based scheme has the following four stages:

1) Split phase: The original signal is divided into two disjoint subsets. Significantly, the variable *Xe* denotes the set of even samples and *Xo* denotes the set of odd samples. This phase is also called lazy wavelet transform because it does not decorrelate the data but only subsamples the signal into even and odd samples.

2) Predict phase: The predicting operator P is applied to the subset *Xo* to obtain the wavelet coefficients d[n] as in (8).

$$d[n]=Xo[n]+P\times(Xe[n]). \tag{8}$$

3) Update phase: *Xe*[n] and d[n] are combined to obtain the scaling coefficients s[n] after an update operator U as in (9).

$$s[n]=Xe[n]+U\times(d[n]). \tag{9}$$

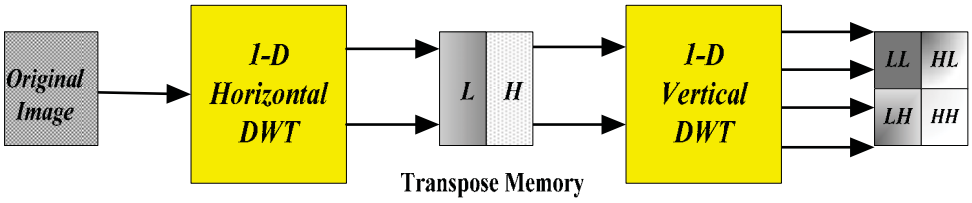
4) Scaling: In the final step, the normalization factor is applied on s[n] and d[n] to obtain the wavelet coefficients. For example, (10) and (11) describe the implementation of the 5/3 integer lifting analysis DWT and are used to calculate the odd (high-pass) and even coefficients (low-pass), respectively.

$$d^*[n] = X(2n+1) - \lfloor X(2n) + X(2n+2)/2 \rfloor. \tag{10}$$

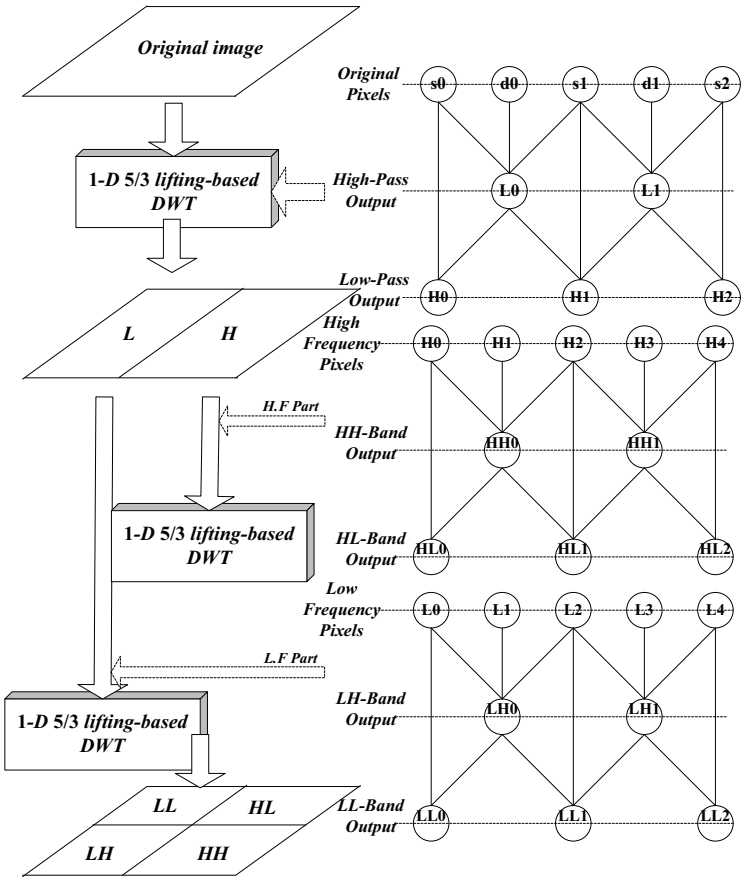
$$s^*[n] = X(2n) - \lfloor d(2n-1) + d(2n+1) + 2/4 \rfloor. \tag{11}$$

Although the lifting-based scheme has low complexity, its long and irregular signal paths cause the major limitation for efficient hardware implementations. Additionally, the increasing number of pipelined registers increases the internal memory size of the 2-D DWT architecture. The 2-D LDWT uses a vertical 1-D LDWT subband decomposition and a horizontal 1-D LDWT subband decomposition to find the 2-D LDWT coefficients. Therefore, the memory requirement dominates the hardware cost and architectural complexity of 2-D LDWT. Fig. 4 shows the 5/3 mode 2-D LDWT operation. The default wavelet filters in JPEG2000 are dual-mode (5/3 and 9/7 modes) LDWT (Taubman & Marcellin, 2001). The lifting-based steps associated with the dual-mode wavelets are shown in Figs. 5 and 6,

respectively. Assuming that the original signals are infinite in length, the first lifting stage is first applied to perform the DWT.



(a)



(b)

Fig. 4. 5/3 mode 2-D LDWT operation. (a) The block diagram flow of a traditional 2-D DWT. (b) Detailed processing flow.

Fig. 5 shows the lifting-based step associated with the wavelet algorithm. The original signals including $s0$, $d0$, $s1$, $d1$, $s2$, $d2$, … are the input pixel sequences. If the original signals are infinite in length, then the first-stage lifting is applied to update the odd index data $s0$, $s1$, …. In (12), the parameters $-1/2$ and H_i denote the first stage lifting parameter and outcome, respectively. Equation (12) shows the operation of the 5/3 integer LDWT (Wu & Lin, 2005) (Martina & Masera, 2007) (Hsia & Chiang, 2008).

$$H_i = [(S_i+S_{i+1})\times\alpha+d_i]\times K_0,$$

$$L_i = [(H_i+H_{i-1})\times\beta+S_i]\times K_1, \tag{12}$$

where $\alpha= -1/2$, $\beta=1/4$, and $K_0= K_1= 1$.

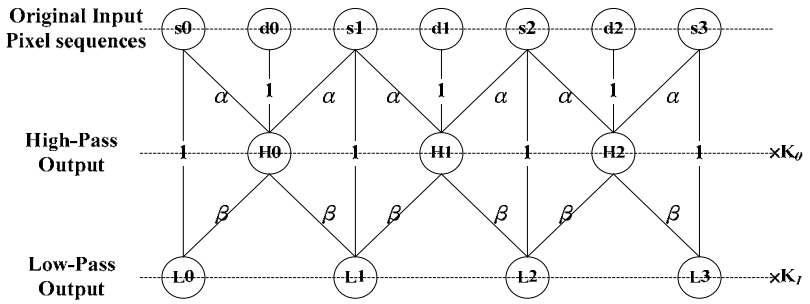


Fig. 5. 5/3 LDWT algorithm.

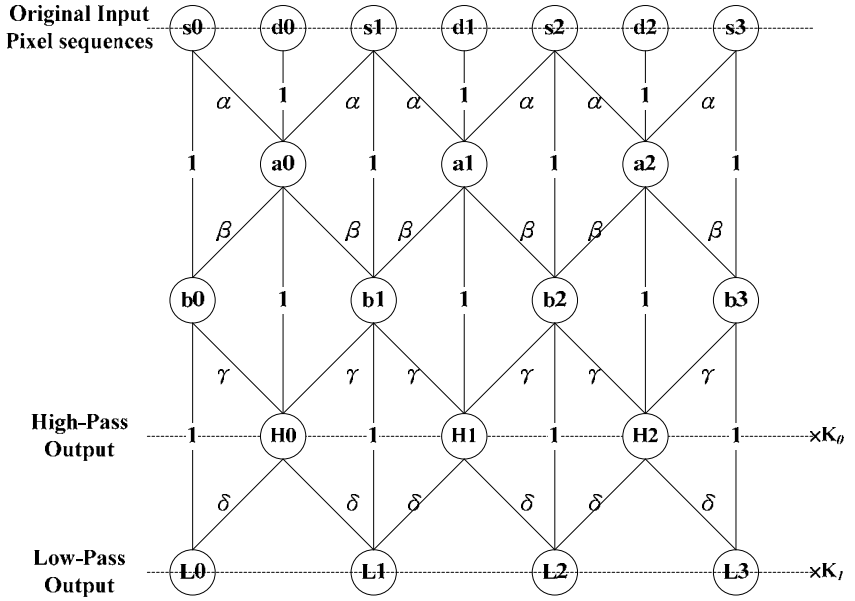


Fig. 6. 9/7 LDWT algorithm.

Together with the high-frequency lifting parameter, α, and the input signal we can find the first stage high-frequency wavelet coefficient, H_i. After H_i is found, H_i together with the low-frequency parameter, β, and the input signals of the second stage low-frequency wavelet coefficients, L_i, can be found. The third and fourth stages lifting can be found in a similar manner.
Similar to the 1-level 1-D 5/3 mode LDWT, the calculation of a 1-level 1-D 9/7 mode LDWT is shown in (13).

$$\begin{aligned} a_i &= [(S_i+S_{i+1})\times\alpha+d_i], \\ b_i &= [(a_i+a_{i-1})\times\beta+S_i], \\ H_i &= [(b_i+b_{i+1})\times\gamma+a_i]\times K_0, \\ L_i &= [(H_i+H_{i-1})\times\delta+b_i]\times K_1. \end{aligned} \tag{13}$$

where α= −1.586134142, β= −0.052980118, γ= +0.882911075, δ= +0.443506852, K_0= 1.230174104, and K_1= 1/K_0.
The calculation comprises four lifting steps and two scaling steps.

3.3 Boundary extension treatment for LDWT

The finite-length signal processed by DWT leads to the boundary effect. The JPEG2000 standard enhances the symmetric extension pixel at the edge, as shown in Table 1. An appropriate signal extension is required to maintain the same number of the wavelet coefficients as in the original signal. The embedded signal extension algorithm (Tan & Arslan, 2001) can be used to compute the boundary of the image.
Since both the extended signal and the lifting structure are symmetrical, all the intermediate and final results of the lifting-based DWT are also symmetrical with regard to the boundary points, and the boundary extension can be performed without additional computational complexity. The inverse lifting structure is easily derived from Table 1 (Tan & Arslan, 2001). The boundary extension signal reflects the signal to i_{left} samples on the left and i_{right} samples on right. Table 1 shows the extension parameters i_{left} and i_{right} for the reversible transform 5/3 mode and the irreversible transform 9/7 mode.

i_0	i_{left}(5/3)	i_1	i_{right}(5/3)	i_0	i_{left}(9/7)	i_1	i_{right}(9/7)
even	2	odd	1	even	4	odd	3

* i_0: first sample index; i_1: last sample index.
Table 1. Boundary extension to the left and to the right for JPEG2000.

4. Interlaced read scan algorithm (IRSA)

In recent years, many 2-D LDWT architectures have been proposed to meet the requirements of on-chip memory for real-time processing. However, the hardware utilization of these architectures needs to be further improved. In DWT implementation, a 1-D DWT needs very massive computation and therefore the computation unit takes most of the hardware cost (Chen & Wu, 2002) (Andra et al., 2000) (Andra et al., 2002) (Diou et al., 2001) (Chen, 2002) (Chiang & Hsia, 2005) (Chen, 2004) (Huang et al., 2005) (Daubechies & Sweldens, 1998) (Marino, 2000) (Vishwanath et al., 1995) (Taubman & Marcellin, 2001)

(Varshney et al., 2007) (Huang et al., 2004) (Tan & Arslan, 2003) (Seo & Kim, 2007) (Huang et al., 2005) (Wu & Lin, 2005) (Lan et al., 2005) (Wu. & Chen, 2001). A 2-D DWT is composed of two 1-D DWTs and a block of transpose memory. In the conventional approach, the size of the transpose memory is equal to the size of the processed image signal. Fig. 7(a) shows the concept of the proposed dual-mode LDWT architecture, which consists of signal arrangement unit, processing element, memory unit, and control unit, as shown in Fig. 7(b). The outputs are fed to the 2-D LDWT four-subband coefficients, HH, HL, LH, and LL. The proposed architecture is described in detail in this section, and we focus on the 2-D dual-mode LDWT.

Compared to the computation unit, the transpose memory becomes the main overhead in the 2-D DWT. The block diagram of a conventional 2-D DWT is shown in Fig. 4. Without loss of generality, the 2-D 5/3 mode LDWT is considered for the description of the 2-D LDWT. If the image dimension is $N \times N$, during the transformation we need a large block of transpose memory (order of N^2) to store the DWT coefficients after the computation of the first stage 1-D DWT decomposition. The second stage 1-D DWT then uses the stored data to compute the 2-D DWT coefficients of the four subbands (Chen & Wu, 2002) (Andra et al., 2000) (Andra et al., 2002) (Diou et al., 2001) (Chen, 2002) (Varshney et al., 2007) (Huang et al., 2004) (Tan & Arslan, 2003) (Seo & Kim, 2007) (Huang et al., 2005) (Wu & Lin, 2005) (Lan et al., 2005) (Wu. & Chen, 2001). The computation and the access of the memory may take time and therefore the latency is long. Since the memory size of N^2 is a large quantity, here we try to use the approach, interlaced read scan algorithm (IRSA), to reduce the required transpose memory to an order of $2N$ or $4N$ (5/3 or 9/7 mode).

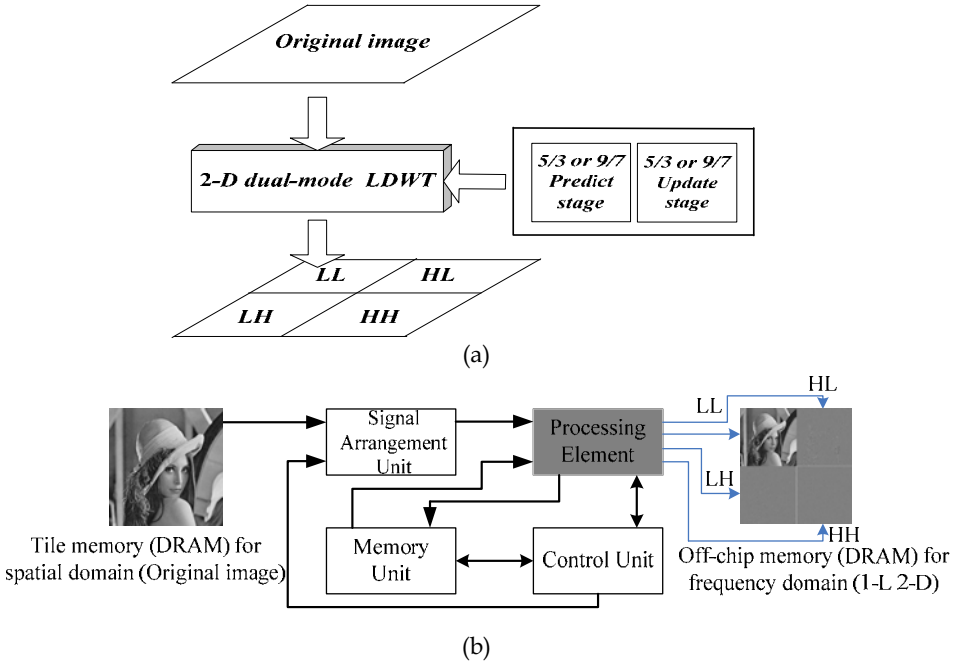
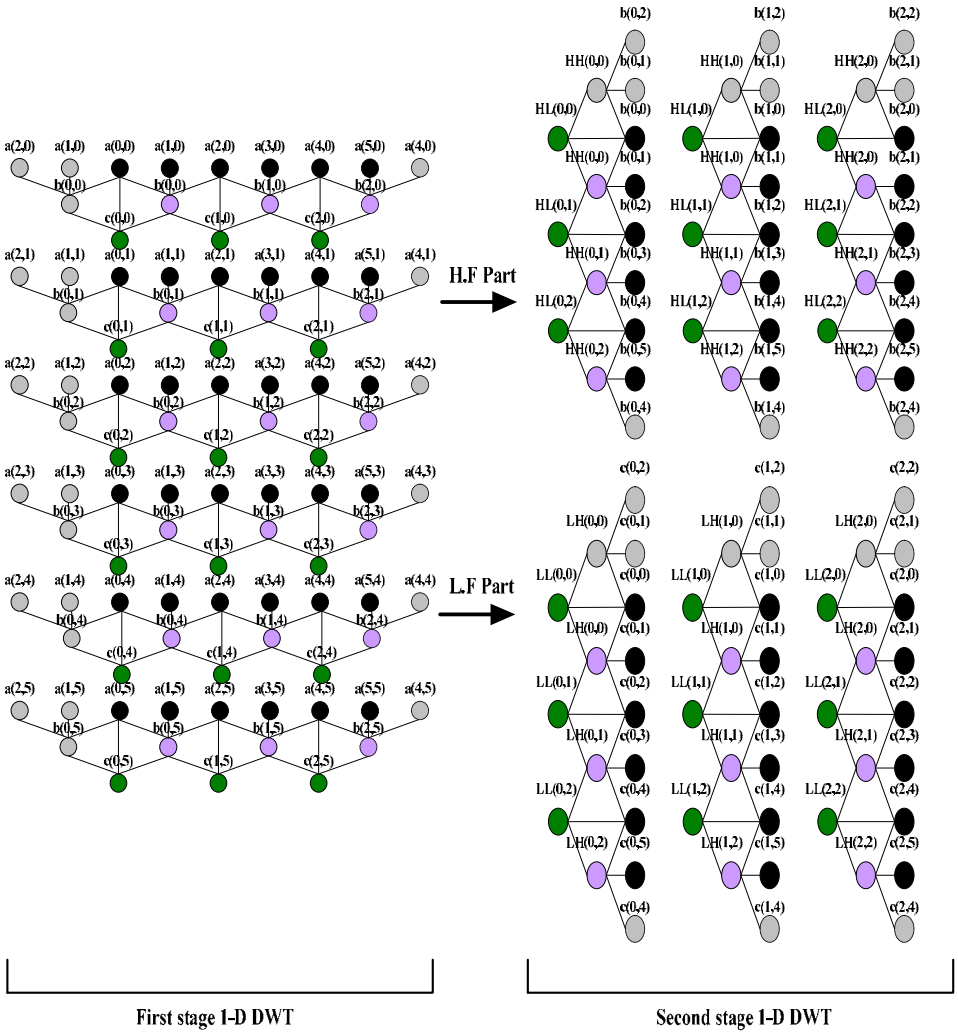


Fig. 7. The system block diagram of the proposed 2-D DWT. (a) 2-D dual-mode LDWT. (b) Block diagram of the proposed system architecture.



$x(i,j)$: original image, i = 0~5 and j = 0~5
$b(i,j)$: high frequency wavelet coefficient of 1-D LDWT
$c(i,j)$: low frequency wavelet coefficient of 1-D LDWT
HH: high-high frequency wavelet coefficient of 2-D LDWT
HL: high-low frequency wavelet coefficient of 2-D LDWT
LH: low-high frequency wavelet coefficient of 2-D LDWT
LL: low-low frequency wavelet coefficient of 2-D LDWT
Fig. 8. Example of 2-D 5/3 mode LDWT operations.

Without loss of generality, let us take a 6×6-pixel image to describe the 2-D 5/3 mode LDWT operation and IRSA. Fig. 8 shows the operation diagram of the 2-D 5/3 mode LDWT operations of a 6×6 image. In Fig. 8, $x(i,j)$, i = 0 to 5 and j = 0 to 5, represents the original

image signal. The left most two columns are the left boundary extension columns, and the right most column is the right boundary extension column. The details of the boundary extension were described in the previous section. The left half of Fig. 8 shows the first stage 1-D DWT operations. The right half of Fig. 8 shows the second stage 1-D DWT operations for finding the four subband coefficients, HH, HL, LH, and LL. In the first stage 1-D DWT, three pixels are used to find a 1-D high-frequency coefficient. For example, $x(0,0)$, $x(0,1)$, and $x(0,2)$ are used to find the high-frequency coefficient $b(0,0)$, $b(0,0) = -[x(0,0) + x(0,2)]/2 + x(0,1)$. To calculate the next high-frequency coefficient $b(0,1)$, we need pixels $x(0,2)$, $x(0,3)$, and $x(0,4)$. Here $x(0,2)$ is used to calculate both $b(0,0)$ and $b(0,1)$ and is called the overlapped pixel. The low-frequency coefficient is calculated using two consecutive high-frequency coefficients and the overlapped pixel. For example, $b(0,0)$ and $b(0,1)$ are together with $x(0,2)$ to find the low-frequency coefficient $c(0,1)$, $c(0,1) = [b(0,0) + b(0,1)]/4 + x(0,2)$. The calculated high-frequency coefficients, $b(i,j)$, and low-frequency coefficients, $c(i,j)$, are then used in the second stage 1-D DWT to calculate the four subband coefficients, HH, HL, LH, and LL.

In the second stage 1-D DWT of Fig. 8, the first HH coefficient, HH(0,0), is calculated by using $b(0,2)$, $b(0,1)$, and $b(0,0)$, $HH(0,0) = -[b(0,0) + b(0,2)]/2 + b(0,1)$. The other HH coefficients can be computed in the same manner using three column consecutive $b(i,j)$ signals. For two column consecutive HH coefficients it has an overlapped $b(i,j)$ signal. For example $b(0,3)$ is the overlapped signal for computing HH(0,0) and HH(0,1). To compute HL coefficients, it needs two column consecutive HH coefficients and an overlapped $b(i,j)$ signal. For example, HL(0,1) is computed from HH(0,0), HH(0,1), and $b(0,3)$, $HL(0,1) = [HH(0,0) + HH(0,1)]/4 + b(0,3)$. The LH coefficients are computed from the $c(i,j)$ signal, and each LH coefficient needs the calculation of three $c(i,j)$ signals. For example, LH(0,1) is computed from $c(0,2)$, $c(0,3)$, and $c(0,4)$, $LH(0,1) = -[c(0,2) + c(0,4)]/2 + c(0,3)$. For two column consecutive LH coefficients it has an overlapped $c(i,j)$ signal. For example, $c(0,3)$ is the overlapped signal for computing LH(0,0) and LH(0,1). To compute LL coefficients, it needs two column consecutive LH coefficients and an overlapped $c(i,j)$ signal. For example, LL(0,1) is computed from LH(0,0), LH(0,1), and $c(0,2)$, $LL(0,1) = [LH(0,0) + LH(0,1)]/4 + c(0,2)$. The detail calculation equations for the four subband coefficients are summarized in the following equations:

$$HH(h,v) = x(2h+1,2v+1)+(1/4)\sum_{s=0}^{1}\sum_{t=0}^{1} x(2h+2s,2v+2t)+(-1/2)\sum_{s=-1}^{2} x(2h+|s|,2v+|-1+s|). \quad (14)$$

$$\begin{aligned} HL(h,v) &= (1/4)[HH(h,v-1)+HH(h,v)]+b(h,2v) \\ &= (1/4)[HH(h,v-1)+HH(h,v)]+(-1/2)[x(2h,2v)+x(2h+2,2v)]+x(2h+1,2v). \end{aligned} \quad (15)$$

$$LH(h,v) = (1/4)[HH(h-1,v)+HH(h,v)]+(-1/2)[x(2h,2v)+x(2h,2v+2)]+x(2h,2v+1). \quad (16)$$

$$\begin{aligned} LL(h,v) &= (1/4)[LH(h,v-1)+LH(h,v)]+c(h,2v) \\ &= (1/4)[LH(h,v-1)+LH(h,v)]+(1/4)[b(h-1,2v)+b(h,2v)]+x(2h,2v) \\ &= (1/4)[LH(h,v-1)+LH(h,v)]+(1/4)[(-1/2)x(2h-2,2v)+x(2h-1,2v)+(-1)x(2h,2v)+x(2h+1,2v)+(-1/2)x(2h+2,2v)]+x(2h,2v). \end{aligned} \quad (17)$$

The parameters in the above equations are defined as follows:

h: horizontal row, *v*: vertical column, *x*: original image, b: high-frequency wavelet coefficient of 1-D DWT, c: low-frequency wavelet coefficient of 1-D DWT, –1/2: Prediction parameter, and 1/4: Updating parameter.

From the description of the operations of the 2-D 5/3 mode LDWT we find that each 1-D high-frequency coefficient, $b(i,j)$, is calculated from three image signals, and one of the image signal is overlapped with the previous $b(i,j)$. The 1-D low-frequency coefficient, $c(i,j)$, is calculated from two row consecutive $b(i,j)$'s and an overlapped pixel. The HH, HL, LH, and LL coefficients are computed from $b(i,j)$'s and $c(i,j)$'s. If we can change the scanning order of the first stage 1-D LDWT and the output order of the second stage 1-D LDWT, during the 2-D LDWT operation we need only to store the $b(i,j)$'s to the transpose memory (First-In First-Out, FIFO size of N) and the overlapped pixels to the internal memory (R4+R9 size of N). For an $N \times N$ image, the transpose memory block can be reduced to only a size of $2N$ as shown in Fig. 9. IRSA is based on this idea, and it can reduce the requirement of the transpose memory significantly. The block diagram of IRSA with several pixels of an image is shown in Fig. 9. In Fig. 9, the numbers on top and left represent the coordinate indexes of a 2-D image. In order to increase the operation speed, IRSA scans two pixels in the consecutive rows a time. IN1 (Initial in $x(0,0)$) and IN2 (Initial in $x(1,0)$) are the scanning inputs at beginning. At the first clock, the system scans two pixels, $x(0,0)$ and $x(1,0)$, from IN1 and IN2, respectively. At the second clock, IN1 and IN2 read pixels $x(0,1)$ and $x(1,1)$, respectively. At clock 3, IN1 and IN2 read pixels $x(0,2)$ and $x(1,2)$, respectively. After IN1 and IN2 have read three pixels, the DWT processor tries to compute two 1-D high-frequency coefficients, $b(0,0)$ and $b(0,1)$, and these two high-frequency coefficients are stored in the transpose memory for the subsequent computation of the low-frequency coefficients. Pixels $x(0,2)$ and $x(1,2)$ are stored in the internal memory for the subsequent computation of the 1-D high-frequency coefficients.

At clock 4, the DWT processor scans pixels on row 2 and row 3, and IN1 and IN2 read pixels $x(2,0)$ and $x(3,0)$, respectively. At clock 5, IN1 and IN2 read pixels $x(2,1)$ and $x(3,1)$, respectively. At clock 6, IN1 and IN2 read pixels $x(2,2)$ and $x(3,2)$, respectively. At this moment the DWT processor tries to compute the two high-frequency coefficients, $b(2,0)$ and $b(3,0)$, upon pixels $x(2,0)$ to $x(2,2)$ and $x(3,0)$ to $x(3,2)$ respectively and these two high-frequency coefficients are stored in the transpose memory for the subsequent computation of the low-frequency coefficients. Pixels $x(2,2)$ and $x(3,2)$ are stored in the internal memory for the subsequent computation of the high-frequency coefficients. Then (at clock 7) the DWT processor scans the subsequent two rows to read three consecutive pixels in each row and compute the high-frequency coefficients. The coefficients are stored in the transpose memory and pixels $x(4,2)$ and $x(5,2)$ are stored in the internal memory. This procedure will continue to read three pixels and compute the high-frequency coefficients and store the coefficients to the transpose memory and store pixels $x(2,j)$ and $x(2,j+1)$ to the internal memory in each row until the last row.

Then the DWT processor scans row 0 and row 1 and makes N1 and N2 read pixels $x(0,3)$ and $x(1,3)$, respectively. At the next clock, IN1 and IN2 read pixels $x(0,4)$ and $x(1,4)$, respectively. The DWT processor uses pixels $x(0,3)$, $x(0,4)$, and $x(0,2)$, which was stored previously, to compute the high-frequency coefficient $b(0,1)$. Simultaneously the DWT processor uses pixels $x(1,3)$, $x(1,4)$, and $x(1,2)$, which was stored previously, to compute the high-frequency coefficients $b(1,1)$. As soon as $b(0,1)$ and $b(1,1)$ are found, $b(0,0)$, $b(0,1)$, and $x(0,2)$ are used to generate the low-frequency coefficient $c(0,1)$; $b(1,0)$, $b(1,1)$, and $x(1,2)$ are

used to generate the low-frequency coefficient $c(1,1)$. The computed high-frequency coefficients are then stored in the transpose memory, and pixels $x(0,4)$ and $x(1,4)$ replace pixels $x(0,2)$ and $x(1,2)$ to be stored in the internal memory. IN1 and IN2 then read the data in rows 2 and 3 to process the same operations until the end of the last pixel. The detail operations are shown in Fig. 10.

The second stage 1-D DWT works in the similar manner as the first stage 1-D DWT. In the HH and HL operations, when three column consecutive $b(i,j)$'s are found in the first stage 1-D DWT, an HH coefficient can be computed. As soon as two column consecutive HH coefficients are found, the two HH coefficients and the overlapped $b(i,j)$'s can be combined to compute an HL coefficient. Similarly, when three column consecutive $c(i,j)$'s are found in the first stage 1-D DWT, an LH coefficient can be computed. As soon as two LH coefficients are found, the two LH coefficients and the overlapped $c(i,j)$ are used to compute an LL coefficient. The detailed operations for the second stage 1-D DWT are shown in Fig. 11.

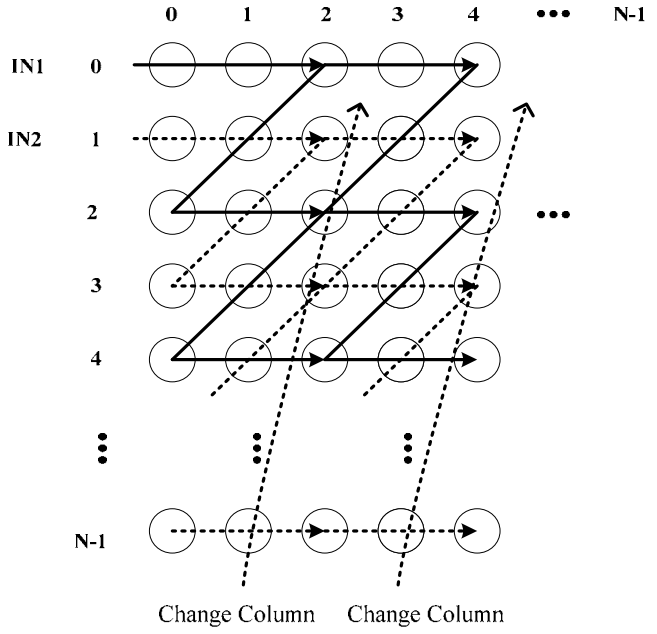


Fig. 9. IRSA of the 2-D LDWT.

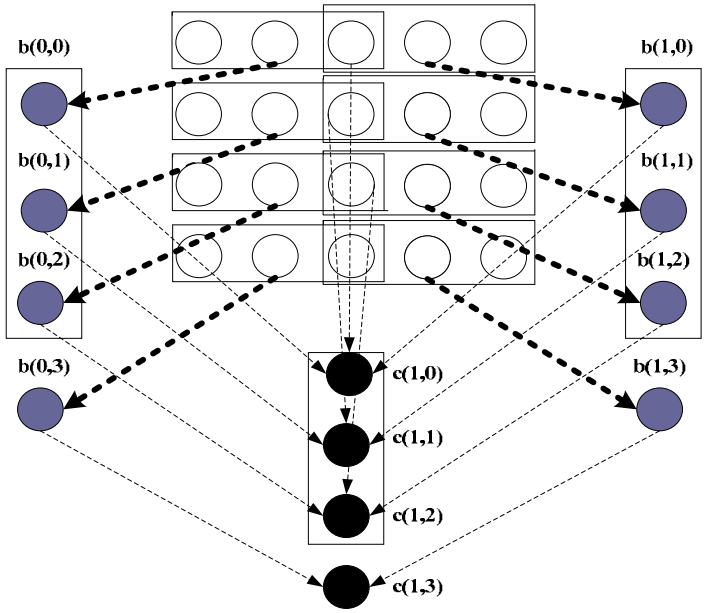
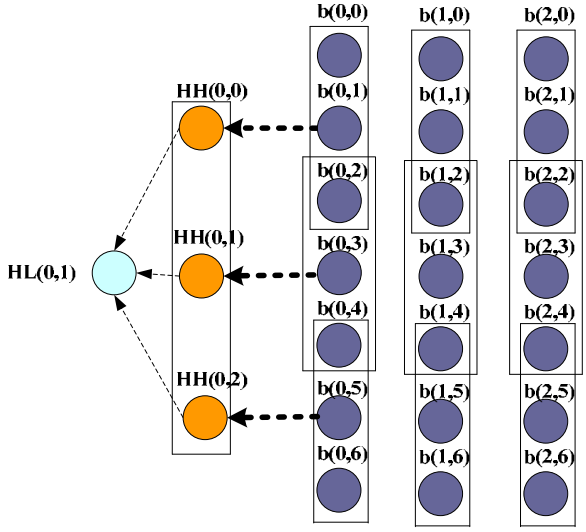
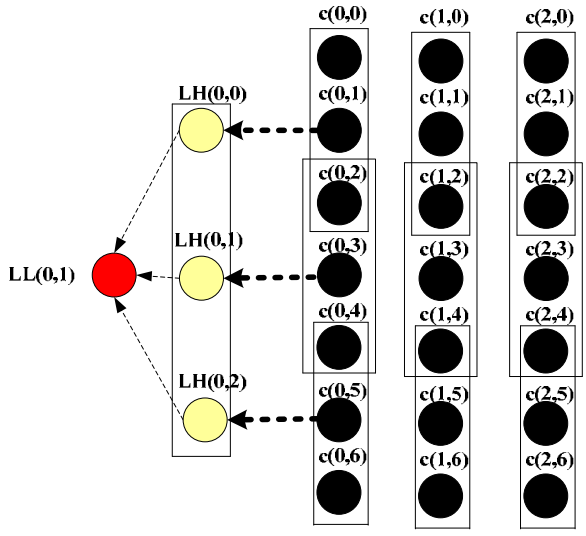


Fig. 10. The detail operations of the first stage 1-D DWT.



(a)



(b)

Fig. 11. The detailed operations of the second stage 1-D DWT. (a) The HF (HH and HL) part operations. (b) The LF (LH and LL) part operations.

5. VLSI architecture and implementation for the 2-D dual-mode LDWT

The IRSA approach has been discussed in the previous section, and the architecture of IRSA is described in this section. We can manipulate the control unit to read off-chip memory. In IRSA, two pixels are scanned concurrently, and the system needs two processing units. For the 2-D LDWT processing, the pixels are processed by the first stage 1-D DWT first. The outputs are then fed to the second stage 1-D DWT to find the four subband coefficients, HH, HL, LH, and LL. There are two parts in the architecture, the first stage 1-D DWT and the second stage 1-D DWT. Here we concentrate on the 2-D 5/3 mode LDWT.

5.1 The first stage 1-D LDWT

The first stage 1-D LDWT architecture consists of the following units: signal arrangement unit, multiplication and accumulation cell (MAC), multiplexer (MUX), and FIFO register. The block diagram is shown in Fig. 12.

The signal arrangement unit consists of three registers, R1, R2, and R3. The pixels are input to R1 first, and subsequently the content of R1 is transferred to R2 and then R3, and R1 keeps reading the following pixels. The operation is like a shift register. As soon as R1, R2, and R3 get signal data, MAC starts operating. The signal arrangement unit is shown in Fig. 13. In Fig. 13 MAC operates at the clock with gray circles.

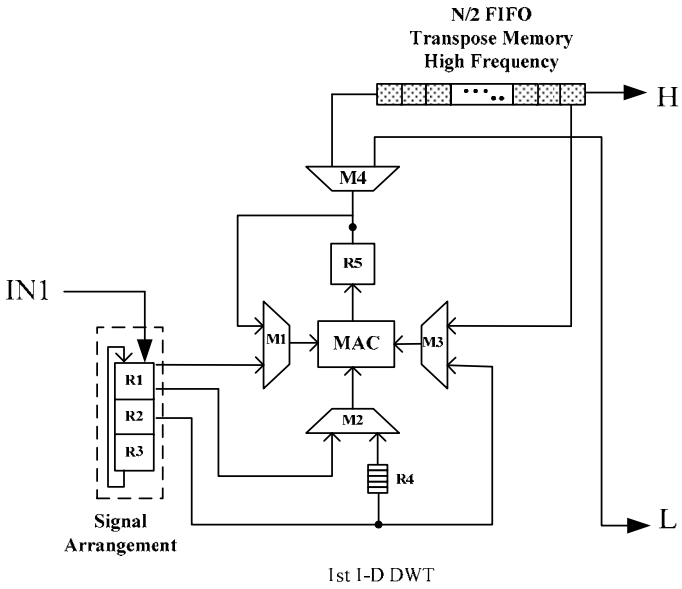


Fig. 12. The architecture of the first stage 1-D DWT.

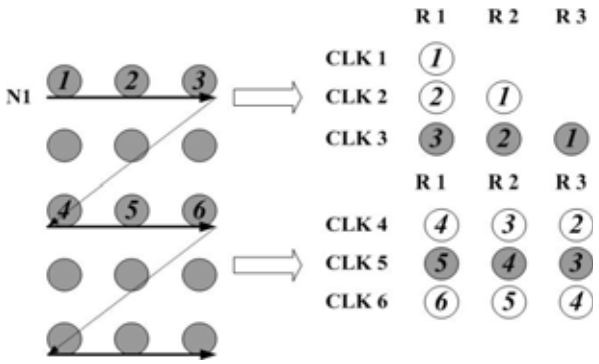


Fig. 13. The operation of the signal arrangement unit (for example, IRAS signal in N1).

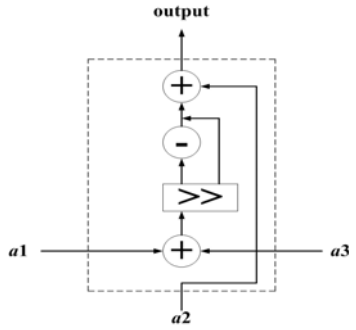


Fig. 14. The block diagram of MAC.

For the low-frequency coefficients calculation we need two high-frequency coefficients and an original pixel. Internal register R4 is used to store the original even pixel (N1) and internal register R9 is used to store the original odd pixel (N2). We can simply shift the content of R3 to R4 after the MAC operation. The FIFO is used to store the high-frequency coefficients to calculate the low-frequency coefficients. Register R5 has two functions: 1) to store the high-frequency coefficients for the low-frequency coefficient calculation, 2) to be used as a signal buffer for MAC. MAC needs time to compute the signal, and the output of MAC cannot directly feed the result to the output or the following operation may be incorrect due to the synchronization problems. R5 acts as an output buffer for MAC to prevent the error in the following operations. In the 5/3 integer lifting-based operations, MAC is used to find the results of the high-frequency output, $-(a1+a3)/2 + a2$, and the low frequency output, $(a1+a3)/4+a2$. There are two multiplication coefficients, –1/2 and 1/4. To save hardware, we can use shifters to implement the –1/2 and 1/4 multiplications. Therefore the MAC needs adders, complementer, and shifters. The MAC block diagram is shown in Fig. 14, where $a1$, $a2$, and $a3$ are the inputs, ''–'' the 2's complement converter, and ''>>'' the right shifter.

5.2 The second stage 1-D LDWT

Similar to the first stage 1-D DWT, the second stage 1-D DWT consists of the following units: signal arrangement unit, MAC, and MUX, as shown in Fig. 15. Due to the parallel architecture, two outputs are generated concurrently from the first stage 1-D DWT, and these two outputs must be merged in the second stage 1-D DWT. The signal arrangement unit processes the signal merging; Fig. 16 shows the processing diagram of the signal arrangement unit. At beginning, signals H0 and H1 are from IN1 and IN2 and these two signals are stored in R3 and R4, respectively. At the next clock, H0 and H1 are moved to R1 and R2 respectively, and concurrently new signals H3 and H4 from IN1 and IN2 are stored to R3 and R4 respectively. The signal arrangement unit operates repeatedly to input signals for the second stage 1-D DWT.

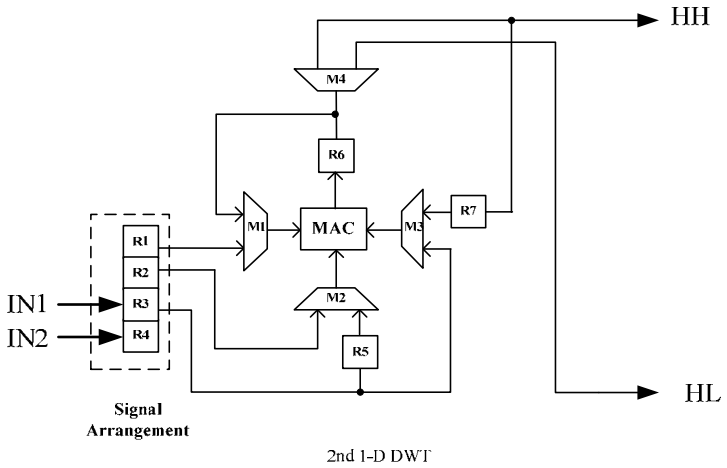


Fig. 15. The block diagram of the second stage 1-D LDWT.

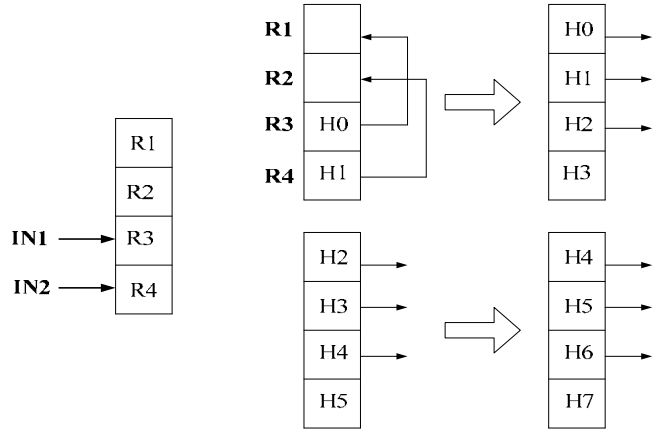


Fig. 16. Signal merging process for the signal arrangement unit.

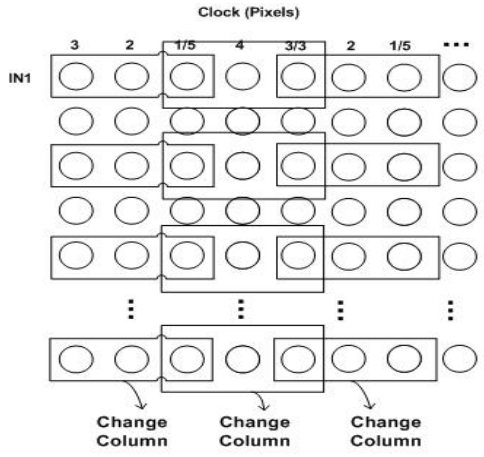
5.3 2-D LDWT architecture

In our IRSA operation, IN1 and IN2 read signals of even row and odd row in a zig-zag order, respectively. The detail is shown in Fig. 17. The block diagram of the proposed 2-D LDWT is shown in Fig. 19(b). It consists of two stages, the first stage 1-D DWT and the second stage 1-D DWT. This architecture needs only a small amount of transpose memory.

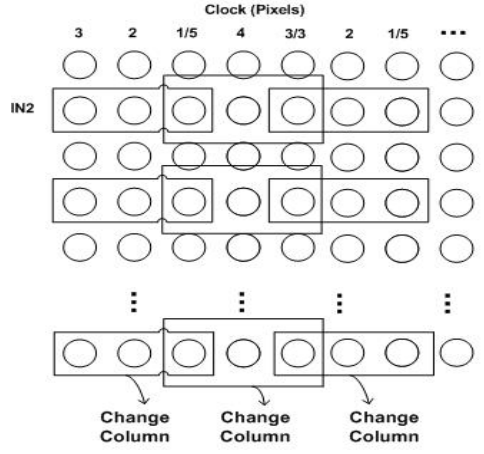
Let us consider a 4×4 image. The signal processing of the first stage 1-D DWT is shown in Fig. 18. The pixels from the even rows are processed by the upper 1-D DWT, and the pixels from the odd rows are processed by the lower 1-D DWT. Each 1-D DWT generates one set of 2×2 high-frequency coefficients and one set of 2×2 low-frequency coefficients, respectively. The generated coefficients are fed to the second stage 1-D DWT under the direction of the arrow head. The input for each second stage 1-D DWT becomes a set of 2×4 signals. The

signal processing of the second stage 1-D DWT is shown in Fig. 19. The 2×4 signals in each second stage 1-D DWT are then processed, and then HH, HL, LH, and LL are generated and each has 2×2 signal data.

The complete architecture of the 2-D LDWT is shown in Fig. 19. The complete 2-D LDWT consists of four parts, two sets of the first stage 1-D DWT, two sets of the second stage 1-D DWT, control unit, and MAC unit.



(a)



(b)

Fig. 17. The input signal sequences. (a) IN1 read signal of even row in zig-zag orders. (b) IN2 read signal of odd row in zig-zag orders.

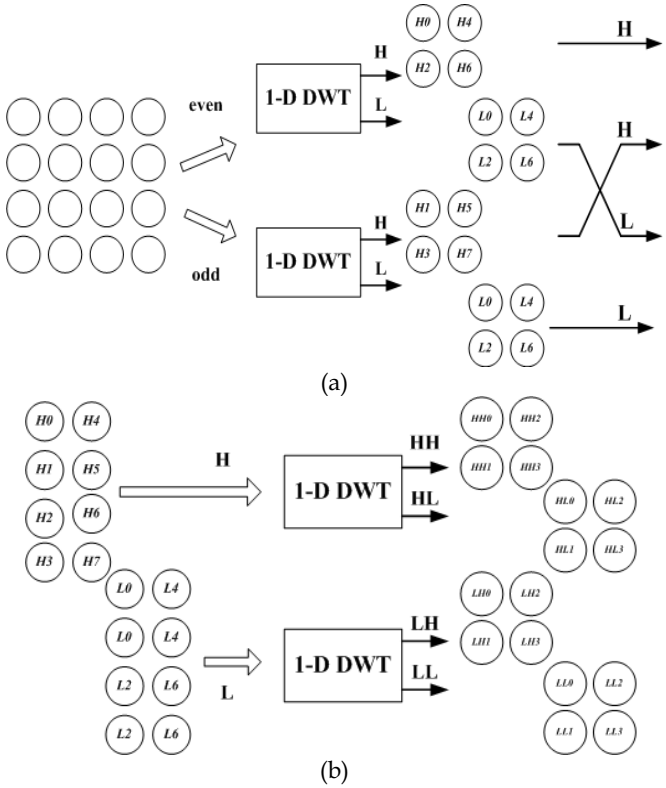


Fig. 18. The signal process of the two stage LDWT. (a) First stage 1-D LDWT. (b) Second stage 1-D LDWT.

According to (12) and (13), the proposed IRSA architecture can also be applied to the 9/7 mode LDWT. Fig. 20 illustrates the approach. From Figs. 10 and 11 in Section 3, the original signals (denoted as black circles) for both 5/3 and 9/7 modes LDWT can be processed by the same IRSA for the first stage 1-D DWT operation. The high-frequency signals (denoted as grey circles) and the correlated low-frequency signals together with the results of the first stage are used to compute the second stage 1-D DWT coefficients. Compared to the 9/7 mode LDWT computation, the 5/3 mode LDWT is much easier for computation, and the registers arrangement in Figs. 12 and 15 is simple. For 9/7 mode LDWT implementation with the same system architecture of 5/3 mode LDWT, we have to do the following modifications: 1) The control signals of the MUX in Figs. 12 and 15 must be modified. We have to rearrange the registers for the MAC block to process the 9/7 parameters. 2) The wavelet coefficients of the dual-mode LDWT are different. The coefficients are $\alpha = -1/2$ and $\beta = 1/4$ for 5/3 mode LDWT, but the coefficients are $\alpha = -1.586134142$, $\beta = -0.052980118$, $\gamma = +0.882911075$, and $\delta = +0.443506852$ for 9/7 mode LDWT. For calculation simplicity and good precision, we can use the integer approach proposed by Huang *et al*. (Huang et al., 2004) and Martina *et al*. (Martina & Masera, 2007) for 9/7 mode LDWT calculation. Similar to the multiplication implementation by shifters and adders in the 5/3 mode LDWT, we can adopt the shifters approach proposed

in (Huang et al., 2005) further to implement the 9/7 mode LDWT. 3) According to the characteristics of the 9/7 mode LDWT, the control unit in Fig. 19(b) must be modified accordingly.

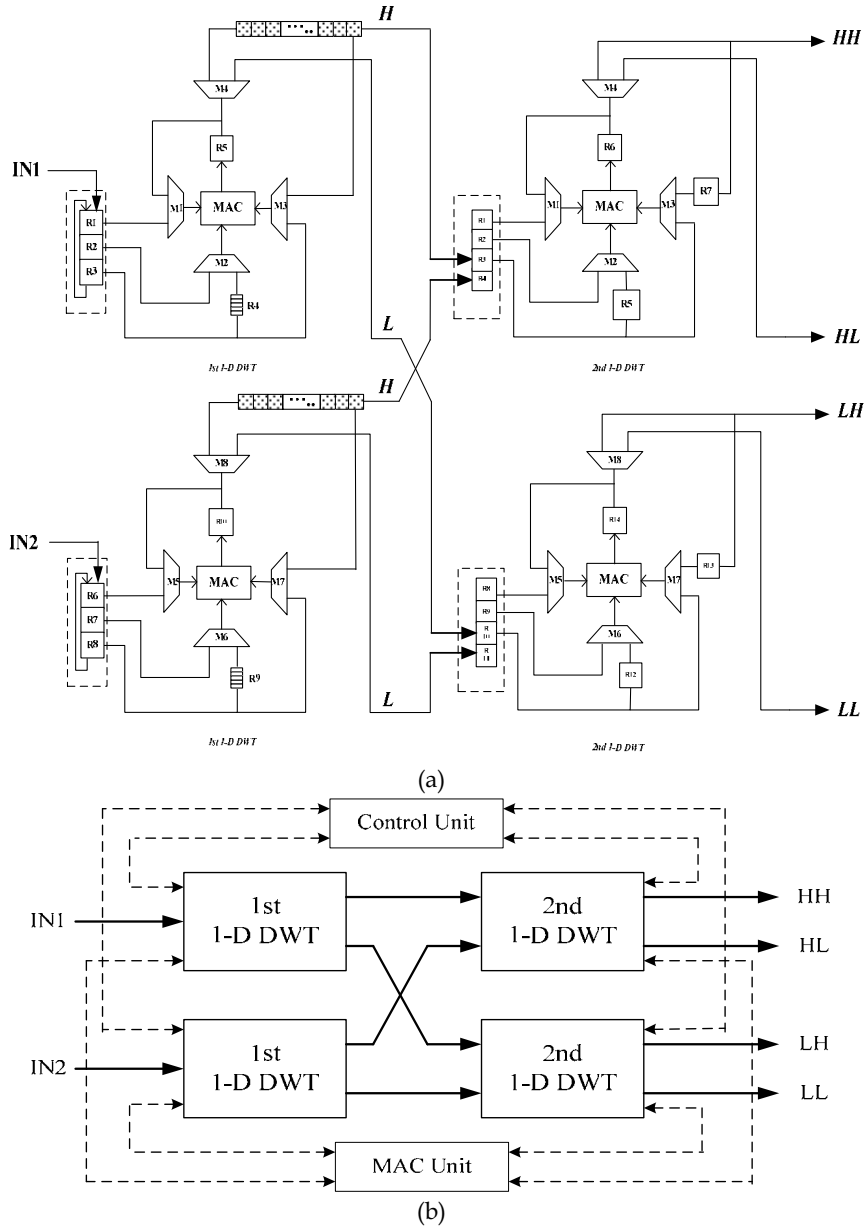


Fig. 19. The complete 2-D DWT block diagram. (a) DSP diagram of the 2-D LDWT. (b) System diagram of the 2-D LDWT.

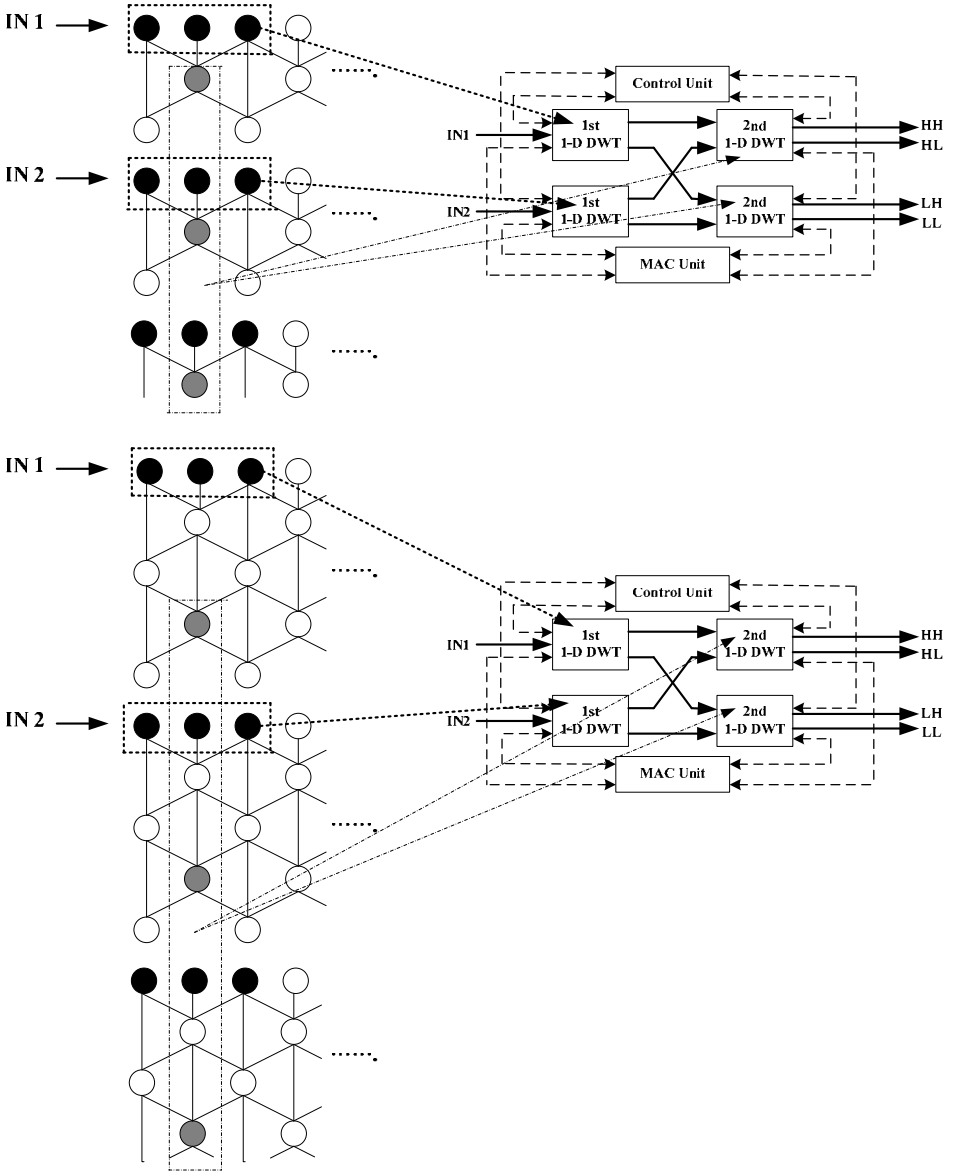


Fig. 20. The processing procedures of 2-D dual-mode LDWTs under the same IRSA architecture.

The multi-level DWT computation can be implemented in a similar manner by the high performance 1-level 2-D LDWT. For the multi-level computation, this architecture needs $N^2/4$ off-chip memory. As illustrated in Fig. 21, the off-chip memory is used to temporarily store the LL subband coefficients for the next iteration computations. The second level

computation requires $N/2$ counters and $N/2$ FIFO's for the control unit. The third level computation requires $N/4$ counters and $N/4$ FIFO's for the control unit. Generally in the jth level computation, we need $N/2^{j-1}$ counters and $N/2^{j-1}$ FIFO's.

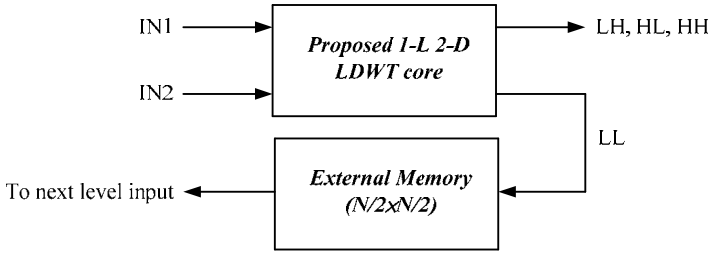


Fig. 21. The multilevel 2-D DWT architecture.

6. Experimental results and comparisons

The 2-D dual-mode LDWT considers a trade-off between low transpose memory and low complexity in the design of VLSI architecture. Tables 2 and 3 show the performance comparisons of the proposed architecture and other similar architectures. Compression results indicate that the proposed VLSI architecture outperforms previous works in terms of transpose memory size, requiring about 50% less memory than the JPEG2000 standard (Chen, 2004) architecture. Moreover, the 2-D LDWT is frame-based, and its implementation bottleneck is the huge transpose memory. Less memory units are needed in our architecture and the latency is fixed on $(3/2)N+3$ clock cycles. Our architecture can also provide an embedded symmetrical extension function. The proposed IRSA approach has the advantages of memory-efficient and high-speed. The proposed 2-D dual-mode LDWT adopts parallel and pipelined schemes to reduce the transpose memory and increase the operation speed. The shifters and adders replace multipliers in the computation to reduce the hardware cost. Chen *et al*. (Chen & Wu, 2002) proposed a folded and pipelined architecture to compute the 2-D 5/3 lifting-based DWT, and they used transpose memory size of $2.5N$ for an $N \times N$ 2-D DWT. This lifting architecture for vertical filtering with two adders and one multiplier is divided into two parts, and each part has one adder and one multiplier. Because both parts are activated in different cycles, they can share the same adder and multiplier. It can increase the hardware utilization and reduce the latency. However, according to the characteristics of the signal flow, it will increase the complexity at the same time.

A 256×256 2-D LDWT was designed and simulated with VerilogHDL and further synthesized by the Synopsys design compiler with TSMC 0.18μm 1P6M CMOS standard process technology. The detailed specs of the 256×256 2-D LDWT are listed in Table 4.

5/3 LDWT architecture	Ours	Diou et al., 2001	Andra et al., 2002	Chen & Wu, 2002	Chen, 2002	Chiang & Hsia, 2005	Mei et al., 2006	Huang et al, 2005	Wu & Lin, 2005
Transpose memory[1] (bytes)	2N	3.5N	3.5N	2.5N	3N	$N^2/4+5N$	2N	3.5N	3.5N
Computation time[2]	$(3/4)N^2+(3/2)N+7$	---	$(N^2/2)+N+5$	N^2	$(N^2/2)+N+5$	N^2	$(N^2/2)+N$	---	$10+(4/3)N^2[1-(1/4)]+2N[1-(1/2)]$
Adders	8	12	8	6	5	4	8	---	---
Multipliers	0	6	4	4	0	0	0	---	6

[1] Transpose memory size is used to store frequency coefficients in the 1-L 2-D DWT.
[2] In a system, computing time represents the time used to compute an image of size $N \times N$.
[3] Suppose the image is of size $N \times N$.
Table 2. Comparisons of 2-D architectures for 5/3 LDWT.

9/7 LDWT architecture	Ours	Andra et al., 2002	Jung & Park, 2005	Chen, 2004[1]	Vishwanath et al., 1995	Huang et al., 2005	Huang et al, 2005	Wu & Lin, 2005	Lan et al., 2005	Wu. & Chen, 2001
Transpose memory (bytes)	4N	N^2	12N	$N^2/4+LN+L$	22N	14N	5.5N	5.5N	---	N^2+4N+4
Computation time	$(3/4)N^2+(3/2)N+7$	$4N^2/3+2$	N^2	$N^2/2\sim(2/3)N$	N^2	---	---	$22+(4/3)N^2[1-(1/4)]+6N[1-(1/2)]$	---	$2N^2/3$
Adders	16	8	12	4 L	36	16	16	8	32	16
Multipliers	0	4	9	4 L	36	12	10	6	20	16

[1] L: the filter length.
Table 3. Comparisons of the 2-D architectures for 9/7 LDWT.

Chip specification	N = 256, Tile size = 256×256
Gate count	29,196 gates
Power supply	1.8V
Technology	TSMC 0.18μm 1P6M (CMOS)
On-Chip memory size (Transpose + Internal)	2-D 5/3 DWT: 512 bytes 2-D 9/7 DWT: 1,024 bytes
Latency	$(3/2)N+3$ = 387 clock cycles
Computing time	$(3/4)N^2+(3/2)N+7$ = 49,543 clock cycles
Maximum clock rate	83 MHz

Table 4. Design specification of the proposed 2-D DWT.

7. Conclusions

This work presents a new architecture to reduce the transpose memory requirement in 2-D LDWT. The proposed architecture has a mixed row- and column-wise signal flow, rather than purely row-wise as in traditional 2-D LDWT. Further we propose a new approach, interlaced read scan algorithm (IRSA), to reduce the transpose memory for a 2-D dual-mode LDWT. The proposed 2-D architectures are more efficient than previous architectures in trading off low transpose memory, output latency, control complexity, and regular memory access sequence. The proposed architecture reduces the transpose memory significantly to a memory size of only 2*N* or 4*N* (5/3 or 9/7 mode) and reduces the latency to (3/2)*N*+3 clock cycles. Due to the regularity and simplicity of the IRSA LDWT architecture, a dual mode (5/3 and 9/7) 256×256 2-D LDWT prototyping chip was designed by TSMC 0.18μm 1P6M standard CMOS technology. The 5/3 and 9/7 filters with different lifting steps are realized by cascading the four modules (split, predict, update, and scaling phases). The prototyping chip takes 29,196 gate counts and can operate at 83 MHz. The method is applicable to any DWT-based signal compression standard, such as JPEG2000, Motion-JPEG2000, MPEG-4 still texture object decoding, and wavelet-based scalable video coding (SVC).

8. References

Andra, K.; Chakrabarti, C. & Acharya, T. (2000). A VLSI architecture for lifting-based wavelet transform, *IEEE Workshop on Signal Processing Systems*, (October 2000) pp. 70-79.

Andra, K.; Chakrabarti, C. & Acharya, T. (2002). A VLSI architecture for lifting-based forward and inverse wavelet transform, *IEEE Transactions on Signal Processing*, Vol. 50, No.4, (April 2002) pp. 966-977.

Chen, P.-Y. (2002). VLSI implementation of discrete wavelet transform using the 5/3 filter, *IEICE Transactions on Information and Systems*, Vol. E85-D, No.12, (December 2002) pp. 1893-1897.

Chen, P.-Y. (2004). VLSI implementation for one-dimensional multilevel lifting-based wavelet transform, *IEEE Transactions on Computer*, Vol. 53, No. 4, (April 2004) pp. 386-398.

Chen, P. & Woods, J. W. (2004). Bidirectional MC-EZBC with lifting implementation, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 14, No. 10, (October 2004) pp. 1183-1194.

Chen, S.-C. & Wu, C.-C. (2002). An architecture of 2-D 3-level lifting-based discrete wavelet transform, *VLSI Design/ CAD Symposium,* (August 2002) pp. 351-354.

Chiang, J.-S. & Hsia, C.-H. (2005). An efficient VLSI architecture for 2-D DWT using lifting scheme, *IEEE International Conference on Systems and Signals*, (April 2005) pp. 528-531.

Christopoulos, C.; Skodras, A. N. & Ebrahimi, T. (2000). The JPEG2000 still image coding system: An overview, *IEEE Trans. on Consumer Electronics*, Vol. 46, No. 4, (November 2000) pp. 1103-1127.

Daubechies, I. & Sweldens, W. (1998). Factoring wavelet transforms into lifting steps, *The Journal of Fourier Analysis and Applications*, Vol. 4, No.3, (1998) pp. 247-269.

Diou, C.; Torres, L. & Robert, M. (2001). An embedded core for the 2-D wavelet transform, *IEEE on Emerging Technologies and Factory Automation Proceedings*, Vol. 2, (October 2001) pp. 179-186.

Habibi, A. & Hershel, R. S. (1974). A unified representation of differential pulse code modulation (DPCM) and transform coding systems, *IEEE Transactions on Communications*, Vol. 22, No. 5, (May 1974) pp. 692-696.

Hsia, C.-H. & Chiang, J.-S. (2008). New memory-efficient hardware architecture of 2-D dual-mode lifting-based discrete wavelet transform for JPEG2000, *IEEE International Conference on Communication Systems*, (November 2008) pp. 766-772.

Huang, C.-T.; Tseng, P.-C. & Chen, L.-G. (2002). Efficient VLSI architecture of lifting-based discrete wavelet transform by systematic design method, *IEEE International Symposium Circuits and Systems*, Vol. 5, (May 2002) pp. 26-29.

Huang, C.-T.; Tseng, P.-C. & Chen, L.-G. (2004). Flipping structure: An efficient VLSI architecture for lifting-based discrete wavelet transform, *IEEE Transactions on Signal Processing*, Vol. 52, No. 4, (April 2004) pp. 1080-1089.

Huang, C.-T.; Tseng, P.-C. & Chen, L.-G. (2005). VLSI architecture for lifting-based shape-adaptive discrete wavelet transform with odd-symmetric filters, *Journal of VLSI Signal Processing Systems*, Vol. 40, No. 2, (June 2005) pp.175-188.

Huang, C.-T.; Tseng, P.-C. & Chen, L.-G. (2005). Analysis and VLSI architecture for 1-D and 2-D discrete wavelet transform, *IEEE Transactions on Signal Processing*, Vol. 53, No. 4, (April 2005) pp. 1575-1586.

Huang, C.-T.; Tseng, P.-C. & Chen, L.-G. (2005). Generic RAM-based architecture for two-dimensional discrete wavelet transform with line-based method, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 15, No. 7, (July 2005) pp. 910-919.

ISO/IEC 15444-1 JTC1/SC29 WG1. (2000). *JPEG 2000 Part 1 Final Committee Draft Version 1.0*, Information Technology.

ISO/IEC JTC1/SC29/WG1 Wgln 1684 (2000). *JPEG 2000 Verification Model 9.0*.

ISO/IEC 15444-1 JTC1/SC29 WG1. (2000). *Motion JPEG2000, ISO/IEC ISO/IEC 15444-3*, Information Technology.

ISO/IEC JTC1/SC29 WG11. (2001), *Coding of Moving Pictures and Audio*, Information Technology.

Jiang, W. & Ortega, A. (2001). Lifting factorization-based discrete wavelet transform based architecture design, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 11, No. 5, (May 2001) pp. 651-657.

Jung, G.-C. & Park, S.-M. (2005). VLSI implement of lifting wavelet transform of JPEG2000 with efficient RPA (recursive pyramid algorithm) realization, *IEICE Transactions on Fundamental*s, Vol. E88-A, No. 12, (December 2005) pp. 3508-3515.

Kondo, H. & Oishi, Y. (2000). Digital image compression using directional sub-block DCT, *International Conference on Communications Technology*, Vol. 1, (August 2000) p p. 985 -992.

Lan, X.; Zheng, N. & Liu, Y. (2005). Low-power and high-speed VLSI architecture for lifting-based forward and inverse wavelet transform, *IEEE Transactions on Consumer Electronics*, Vol. 51, No. 2, (May 2005) pp. 379-385.

Li, W.-M.; Hsia, C.-H. & Chiang, J.-S. (2009). Memory-efficient architecture of 2-D dual-mode lifting scheme discrete wavelet transform for Moion-JPEG2000, *IEEE International Symposium on Circuits and Systems*, (May 2009) pp. 750-753.

Lian, C.-J.; Chen, K.-F.; Chen, H.-H. & Chen, L.-G. (2001). Lifting based discrete wavelet transform architecture for JPEG2000, *IEEE International Symposium on Circuits and Systems*, Vol. 2, (May 2001) pp. 445-448.

Mallat, S. G. (1989). A theory for multi-resolution signal decomposition: The wavelet representation, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 7, (July 1989) pp. 674-693.

Mallat, S. G. (1989). Multi-frequency channel decompositions of images and wavelet models, IEEE *Transactions on* Acoustics, Speech and Signal Processing, Vol. ASSP-37, No. 12, (December 1989) pp. 2091-2110.

Marcellin, M. W.; Gormish, M. J. & Skodras, A. N. (2000). JPEG2000: The new still picture compression standard, *ACM Multimedia Workshops*, (September 2000) pp. 45-49.

Marino, F. (2000). Efficient high-speed/low-power pipelined architecture for the direct 2-D discrete wavelet transform, *IEEE Transactions on Circuits and Systems II*, Vol. 47, No. 12, (December 2000) pp. 1476-1491.

Martina, M. & Masera, G. (2007). Folded multiplierless lifting-based wavelet pipeline, *IET Electronics Letters*, Vol. 43, No. 5, (March 2007) pp. 27-28.

Mei, K.; Zheng, N. & van de Wetering, H. (2006). High-speed and memory-efficient VLSI design of 2-D DWT for JPEG2000, *IET Electronics Letter*, Vol. 42, No. 16, (August 2006) pp. 907-908.

Ohm, J.-R. (2005). Advances in scalable video coding, *Proceedings of The IEEE,* Invited Paper, Vol. 93, No.1, pp. 42-56, (January 2005) pp. 42-56.

Richardson, I. (2003). *H.264 and MPEG-4 Video Compression*, John Wiley & Sons Ltd.

Seo, Y.-H. & Kim, D.-W. (2007). VLSI architecture of line-based lifting wavelet transform for Motion JPEG2000, *IEEE Journal of Solid-State Circuits*, Vol. 42, No. 2, (February 2007) pp. 431-440.

Sweldens, W. (1996). The lifting scheme: A custom-design construction of biorthogonal wavelets, *Applied and Computation Harmonic Analysis*, Vol. 3, No. 15, (1996) pp.186-200.

Tan, K.C.B. & Arslan, T. (2001). Low power embedded extension algorithm for the lifting based discrete wavelet transform in JPEG2000, *IET Electronics Letters*, Vol. 37, No. 22, (October 2001) pp.1328-1330.

Tan, K.C.B. & Arslan, T. (2003). Shift-accumulator ALU centric JPEG 2000 5/3 lifting based discrete wavelet transform architecture, *IEEE International Symposium on Circuits and Systems,* Vol. 5, (May 2003) pp. V161-V164.

Taubman, D. & Marcellin, M. W. (2001). *JPEG2000 image compression fundamentals, standards, and practice*, Kluwer Academic Publisher.

Varshney, H.; Hasan, M. & Jain, S. (2007). Energy efficient novel architecture for the lifting-based discrete wavelet transform, *IET Image Process*, Vol. 1, No. 3, (September 2007) pp.305-310.

Vishwanath, M.; Owens, R. M. & Irwin, M. J. (1995). VLSI architecture for the discrete wavelet transform, *IEEE Transactions on Circuits and Systems II*, Vol. 42, No. 5, (May 1995) pp. 305-316.

Weeks, M. & Bayoumi, M. A. (2002). Three-dimensional discrete wavelet transform architectures, *IEEE Transactions on Signal Processing*, Vol. 50, Vo.8, (August 2002) pp. 2050-2063.

Wu, B.-F. & Lin, C.-F. (2005). A high-performance and memory-efficient pipeline architecture for the 5/3 and 9/7 discrete wavelet transform of JPEG2000 codec, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 15, No. 12, (December 2005) pp. 1615-1628.

Wu, P.-C. & Chen, L.-G. (2001). An efficient architecture for two-dimensional discrete wavelet transform, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 11, No. 4, (April 2001) pp. 536-545.

VLSI

Edited by Zhongfeng Wang

ISBN 978-953-307-049-0

Hard cover, 456 pages

Publisher InTech

Published online 01, February, 2010

Published in print edition February, 2010

The process of Integrated Circuits (IC) started its era of VLSI (Very Large Scale Integration) in 1970's when thousands of transistors were integrated into one single chip. Nowadays we are able to integrate more than a billion transistors on a single chip. However, the term "VLSI" is still being used, though there was some effort to coin a new term ULSI (Ultra-Large Scale Integration) for fine distinctions many years ago. VLSI technology has brought tremendous benefits to our everyday life since its occurrence. VLSI circuits are used everywhere, real applications include microprocessors in a personal computer or workstation, chips in a graphic card, digital camera or camcorder, chips in a cell phone or a portable computing device, and embedded processors in an automobile, et al. VLSI covers many phases of design and fabrication of integrated circuits. For a commercial chip design, it involves system definition, VLSI architecture design and optimization, RTL (register transfer language) coding, (pre- and post-synthesis) simulation and verification, synthesis, place and route, timing analyses and timing closure, and multi-step semiconductor device fabrication including wafer processing, die preparation, IC packaging and testing, et al. As the process technology scales down, hundreds or even thousands of millions of transistors are integrated into one single chip. Hence, more and more complicated systems can be integrated into a single chip, the so-called System-on-chip (SoC), which brings to VLSI engineers ever increasingly challenges to master techniques in various phases of VLSI design. For modern SoC design, practical applications are usually speed hungry. For instance, Ethernet standard has evolved from 10Mbps to 10Gbps. Now the specification for 100Mbps Ethernet is on the way. On the other hand, with the popularity of wireless and portable computing devices, low power consumption has become extremely critical. To meet these contradicting requirements, VLSI designers have to perform optimizations at all levels of design. This book is intended to cover a wide range of VLSI design topics. The book can be roughly partitioned into four parts. Part I is mainly focused on algorithmic level and architectural level VLSI design and optimization for image and video signal processing systems. Part II addresses VLSI design optimizations for cryptography and error correction coding. Part III discusses general SoC design techniques as well as other application-specific VLSI design optimizations. The last part will cover generic nano-scale circuit-level design techniques.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Chih-Hsien Hsia and Jen-Shiun Chiang (2010). Memory-Efficient Hardware Architecture of 2-D Dual-Mode Lifting-Based Discrete Wavelet Transform for JPEG2000, VLSI, Zhongfeng Wang (Ed.), ISBN: 978-953-307-049-0, InTech, Available from: http://www.intechopen.com/books/vlsi/memory-efficient-hardware-architecture-of-2-d-dual-mode-lifting-based-discrete-wavelet-transform-for

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821